February 2021

METHOD AND SYSTEM FOR MANAGING DATA FLOW BETWEEN SOURCE SYSTEM AND TARGET SYSTEM

Shreyas Kunjal Chandrahas
Visa

METHOD AND SYSTEM FOR MANAGING DATA FLOW BETWEEN SOURCE SYSTEM AND TARGET SYSTEM

VISA

INVENTOR:

Shreyas Kunjal Chandrahas

TECHNICAL FIELD

[0001] This disclosure relates to a method and a system for managing data flow between source system and target system.

BACKGROUND

[0001] In current Extract Transform Load (ETL) pipeline, transferring data from source system to target system requires query to be written manually by an ETL developer. In scenarios, where platform and/or technology supported by the source system and target system are different, a query is to be written manually considering a compatibility between the source system and the target system, as the source and the target format are different. For example, when a job needs data transfer from a HIVE system to a MySQL system, the job needs to be carefully developed to convert the data from the HIVE to the MySQL format.

[0002] Further, writing queries for converting data from source format to target format, requires knowledge and experience of the ETL developer. This may be cumbersome when a significant number of records, such as thousands or lakhs of records, need to be transferred between the disparate systems. Additionally, writing manually the queries for transfer of large quantities of records between disparate systems may create errors, and affect further execution of processes in the ETL pipeline. Resolving such errors, need additional manual effort of the ETL developer for identifying properly errors caused due to compatibility issues of disparate systems or performing parity check for all the records transferred between the source system and target system. In this context, repeated manual intervention may degrade efficiency of the ETL system in terms of increased time and cost associated with the data transfer process.

[0003] More particularly, the existing systems do not provide any efficient mechanism for automatically instantiating a pipeline strategy associated with data transfer based on platform of the source system and platform of the target system. Further, they do not provide any metadata driven data transfer capability between disparate systems, rather requires jobs to be manually developed. Also, the existing systems do not ensure whether good quality data is being transferred with respect to the source data read in accordance with the target data.

[0004] In order to overcome the above-mentioned shortcomings, the present invention provides a meta data driven scalable, distributed point to point data logistic system between disparate data sources along with quality assurance associated with data being transferred.

BRIEF DESCRIPTION OF THE DRAWINGS

[0005] Additional advantages and details are explained in greater detail below with reference to the exemplary embodiments that are illustrated in the accompanying schematic figures, in which:

[0006] **Fig.1** illustrates exemplary architecture for managing data flow from multiple source systems to multiple target systems in accordance with some embodiments of the present disclosure.

[0007] **Fig.2a** illustrates an internal architecture of a data flow management system in accordance with some embodiments of the present disclosure.

[0008] **Fig.2b** illustrates a workflow of the data flow management system in accordance with some embodiments of the present disclosure.

[0009] **Fig.3** illustrates a block diagram of an exemplary computer system for implementing embodiments consistent with the present disclosure.

DESCRIPTION OF THE DISCLOSURE

[0010] It is to be understood that the present disclosure may assume various alternative variations and step sequences, except where expressly specified to the contrary. It is also to be understood that the specific devices and processes illustrated in the attached drawings and described in the following specification are simply exemplary and non-limiting embodiments or aspects. Hence, specific dimensions and other physical characteristics related to the embodiments or aspects disclosed herein are not to be considered as limiting.

[0011] For purposes of the description hereinafter, the terms “end,” “upper,” “lower,” “right,” “left,” “vertical,” “horizontal,” “top,” “bottom,” “lateral,” “longitudinal,” and

derivatives thereof shall relate to the disclosed subject matter as it is oriented in the drawing figures. However, it is to be understood that the disclosed subject matter may assume various alternative variations and step sequences, except where expressly specified to the contrary. It is also to be understood that the specific devices and processes illustrated in the attached drawings, and described in the following specification, are simply exemplary embodiments or aspects of the disclosed subject matter. Hence, specific dimensions and other physical characteristics related to the embodiments or aspects disclosed herein are not to be considered as limiting unless otherwise indicated.

[0012] No aspect, component, element, structure, act, step, function, instruction, and/or the like used herein should be construed as critical or essential unless explicitly described as such. Also, as used herein, the articles "a" and "an" are intended to include one or more items and may be used interchangeably with "one or more" and "at least one." Furthermore, as used herein, the term "set" is intended to include one or more items (e.g., related items, unrelated items, a combination of related and unrelated items, and/or the like) and may be used interchangeably with "one or more" or "at least one." Where only one item is intended, the term "one" or similar language is used. Also, as used herein, the terms "has," "have," "having," or the like are intended to be open-ended terms. Further, the phrase "based on" is intended to mean "based at least partially on" unless explicitly stated otherwise.

[0013] As used herein, the terms "communication" and "communicate" may refer to the reception, receipt, transmission, transfer, provision, and/or the like of information (e.g., data, signals, messages, instructions, commands, and/or the like). For one unit (e.g., a device, a system, a component of a device or system, combinations thereof, and/or the like) to be in communication with another unit means that the one unit is able to directly or indirectly receive information from and/or transmit information to the other unit. This may refer to a direct or indirect connection (e.g., a direct communication connection, an indirect communication connection, and/or the like) that is wired and/or wireless in nature. Additionally, two units may be in communication with each other even though the information transmitted may be modified, processed, relayed, and/or routed between the first and second unit. In some non-limiting

embodiments or aspects, a message may refer to a network packet (e.g., a data packet and/or the like) that includes data. It will be appreciated that numerous other arrangements are possible.

[0014] As used herein, the term "computing device" may refer to one or more electronic devices that are configured to directly or indirectly communicate with or over one or more networks. A computing device may be a mobile or portable computing device, a desktop computer, a server, and/or the like. Furthermore, the term "computer" may refer to any computing device that includes the necessary components to receive, process, and output data, and normally includes a display, a processor, a memory, an input device, and a network interface. A "computing system" may include one or more computing devices or computers. An "application" or "application program interface" (API) refers to computer code or other data sorted on a computer-readable medium that may be executed by a processor to facilitate the interaction between software components, such as a client-side front-end and/or server-side back-end for receiving data from the client. An "interface" refers to a generated display, such as one or more graphical user interfaces (GUIs) with which a user may interact, either directly or indirectly (e.g., through a keyboard, mouse, touchscreen, etc.). Further, multiple computers, e.g., servers, or other computerized devices, such as an autonomous vehicle including a vehicle computing system, directly or indirectly communicating in the network environment may constitute a "system" or a "computing system".

[0015] It will be apparent that systems and/or methods, described herein, can be implemented in different forms of hardware, software, or a combination of hardware and software. The actual specialized control hardware or software code used to implement these systems and/or methods is not limiting of the implementations. Thus, the operation and behavior of the systems and/or methods are described herein without reference to specific software code, it being understood that software and hardware can be designed to implement the systems and/or methods based on the description herein.

[0016] Some non-limiting embodiments or aspects are described herein in connection with thresholds. As used herein, satisfying a threshold may refer to a value being greater than the threshold, more than the threshold, higher than the threshold, greater than or

equal to the threshold, less than the threshold, fewer than the threshold, lower than the threshold, less than or equal to the threshold, equal to the threshold, etc.

[0017] **Fig.1** illustrates exemplary architecture for managing data flow from multiple source systems to multiple target systems in accordance with some embodiments of the present disclosure.

[0018] As shown in **Fig.1**, the architecture 100 may include multiple source systems, a data flow management system 103 and multiple target systems. Here, platform of each of the source systems may be same or different from platform of the target systems. As an example, platforms of source systems and platforms of target systems may include, but not limited to HIVE, MySQL, DB2, Oracle, and like.

[0019] As an example, a job may need to transfer N number of data records 111, 115 from a source system 101 with HIVE platform to a target system 105 with MySQL platform. Here, source format 107 of input dataset may be HIVE and target format 109 of output dataset may be MySQL. To transfer N number of data records 111, 115, the data flow management system 103 may identify source type as HIVE and target type as MySQL based on metadata of the input dataset and the output dataset. Further, the data flow management system 103 may determine transfer type, which may include one of 'REPLACE', 'UPDATE', and 'APPEND'. As an example, when the transfer type is 'REPLACE', the system may replace all N number of data records 111, 115 of HIVE format from the source system 101 with MySQL format for the target system 105. Further, the data flow management system 103 may determine data quality strategy, which may include one of 'FULL_DIFF', and 'COUNT'. In 'FULL_DIFF' data quality strategy, the data flow management system 103 may perform cell to cell comparison of the N number of records. As an example, if three records 111, 115, each having five columns, have to be transferred from source system 101 to target system 105, then data flow management system 103 may compare total of fifteen cells associated with data points to check whether all the fifteen datapoints of the target system 105 are in parity with the fifteen datapoints of the source system 101. However, in 'COUNT' data quality strategy, the data flow management system 103 may compare only total number of records 111, 115 to be transferred. As an example, if three records 111, 115 have to

be transferred from source system 101 to target system 105, then data flow management system 103 may compare three records 111, 115 to check whether all the three records 113, 117 of the target system 105 are in parity with the three records 111, 115 of the source system 101. Further, the data flow management system 103 may automatically generate a pipeline within a NiFi cluster based on the source type, the target type, the transfer type and the data quality strategy. As an example, for the source type HIVE, the target type MySQL, transfer type 'REPLACE' and 'COUNT' data quality strategy, Apache NiFi template may be HIVE-MYSQL-REPLACE-COUNT. Further, the data flow management system 103 may pick the appropriate connection in a NiFi cluster for the pipeline. The data flow management system 103 may also facilitate inbuilt data quality and checkpointing to determine completeness and correctness of data being transferred.

[0020] **Fig.2a** illustrates an internal architecture of a data flow management system in accordance with some embodiments of the present disclosure.

[0021] In an embodiment, the data flow management system 103 may comprise a Metadata Definer (MD) 201, a Data Flow Manager (DFM) 203, and a JFrog Artifactory 205. Here, the MD 201 may define metadata of source dataset, and target dataset. From dataset perspective, the metadata may include, but not limited to name of the table, column names, column types, column default values, precision, scale and table's platform such as HIVE, MySQL, DB2 and the like. Further, the MD 201 may define metadata of data transfer pipeline. From pipeline perspective, the metadata may include, but not limited to mapping of the column from source to target, pipeline frequency, transfer type, and data quality strategy. The MD 201 may input the metadata associated with source system 101 and target system 105 to the DFM 203.

[0022] Further, the DFM 203 may receive metadata from the MD 201, based on which the DFM 203 may determine Source Type, and Target Type from predefined platforms such as Hive, MySQL, DB2, Oracle and the like. Further, the DFM 203 may determine Transfer type from among 'REPLACE', 'UPDATE', 'APPEND' and the like. The DFM 203 may also determine Data Quality strategy to be used in the pipeline from among 'FULL_DIFF', 'COUNT' and the like. Further, the DFM 203 may determine

the frequency of the pipeline to run for transferring data records 111, 115 from the source system 101 to target system 105, for example, daily at 2 AM. The DFM 203 may determine column mapping of data records 111, 115 from source format 107 to target format 109. The DFM 203 may also generate the query to be run on the source system 101 with respect to the target columns from the source system 101. Thereafter, the DFM 203 may select an appropriate Apache NiFi template to be used for instantiating a pipeline with the template in a NiFi cluster to accomplish transfer use case. Here, the DFM 203 may select the appropriate connection in the NiFi cluster for the instantiated pipeline. Apache Ni-Fi is an open-source tool, which may provide a canvas, where a programmer can drag and drop processors, for performing a dedicated job. Further, the DFM 203 may utilize check pointing and data quality-services 211, to track progress of the job or to verify if data transfer process is stuck or not going on as desired. The check pointing and data quality-services 211 may ensure completeness and correctness of data. Here, upon completion of the pipeline, the NiFi pipeline may push checkpoint to checkpoint microservice with the completion checkpoint.

[0023] Further, the JFrog Artifactory 205 may store NiFi templates of all combinations of source types, target types, transfer types, and DQ strategies. As an example, the JFrog Artifactory 205 may store NiFi template HIVE-MYSQL-REPLACE-COUNT to replace data records 111, 115 in HIVE format with MYSQL format and compare total number of records 113, 117 transferred. The NiFi templates may be stored so that the DFM 203 may instantiate the specific pipeline with the template in the Nifi cluster utilizing the JFrog Artifactory 205.

[0024] In an embodiment, the data flow management system 103 may further comprise a microservice mesh which may provide a platform for hosting code associated with the pipeline by the DFM 203. Further, the microservice mesh may instantiate the pipeline in the Nifi cluster. The microservice mesh may also have a persistence layer to store the state of the pipeline.

[0025] In an embodiment, the data flow management system 103 may further comprise a Nifi cluster which may perform transfer of data record from the source system 101 to target system 105 based on the pipeline created by the DFM 203. The Nifi cluster may

have an internal scheduler which may activate the pipeline based on the frequency defined by a pipeline owner. The NiFi may be built as a cluster to have a fault tolerant model with a Distributed file system (DFS) as its file persistence layer. Further, NiFi Registry may synchronize the templates from the JFrog Artifactory 205 with the Nifi cluster.

[0026] **Fig.2b** illustrates a workflow of the data flow management system in accordance with some embodiments of the present disclosure.

[0027] In an embodiment, the MD 201 may define the metadata of the input dataset with source format 107, output dataset with the target format 109 and the data transfer pipeline between the datasets and its internal transformation. The MD 201 may be one of open source or proprietary tool. As illustrated in **Fig.2b**, to perform the pipeline propagation, the MD 201 may input the aforesaid information to the DFM 203.

[0028] Further, the DFM 203 may extract the metadata of input dataset and determine the Source Type to be one of HIVE, MySQL, DB2, Oracle and the like based on the input provided by the MD 201. As an example, the metadata of input dataset may be represented as follows:

App1
T(Table Name) {Hive}
'A' string
'B' int
'C' double

Here, the DFM 203 may determine that application 'App1' has a table 'T' defined in the MD 201. The characters mentioned within the curly brackets may be determined as Source Type, for example Hive. Further, the table may have three columns namely 'A', 'B', and 'C' with string, integer and double data types respectively. The DFM 203 may determine these columns need to be transferred from respective data types with HIVE format.

[0029] Further, the DFM 203 may extract the metadata of output dataset and determine the Target Type to be one of HIVE, MySQL, DB2, Oracle and the like based on the

input provided by the MD 201. As an example, the metadata of output dataset may be represented as follows:

App2
TA(Table Name) {MySQL}
'X' int
'Y' int
'Z' double

[0030] Here, the DFM 203 may determine that application 'App2' has a table 'TA' defined in the MD 201. The characters mentioned within the curly brackets may be determined as Target Type, for example MySQL. Further, the table may have three columns namely 'X', 'Y', and 'Z' with integer, integer and double data types respectively. The DFM 203 may determine the columns 'A', 'B', and 'C' need to be transferred from respective data types with HIVE format to columns 'X', 'Y', and 'Z' with respective data types with MySQL format.

[0031] Further, the DFM 203 may determine the transfer type from among 'REPLACE', 'UPDATE', 'APPEND' and the like. As an example, based on the metadata of the pipeline, the DFM 203 may determine the transfer type as 'REPLACE'. Further, the DFM 203 may determine the Data Quality (DQ) strategy to be used in the pipeline from among 'FULL_DIFF', 'COUNT' and the like. As an example, based on the metadata of the pipeline, the DFM 203 may determine the DQ strategy as 'COUNT'. Further, the DFM 203 may determine frequency of the pipeline to run, as an example daily at 2 AM. Further, the DFM 203 may determine column mapping from source system 101 to target system 105. As an example, the dataset of column 'A' need to be transferred to 'X', the dataset of column 'B', and 'C' need to be transferred to 'Y', and the dataset of column 'C' need to be transferred to 'Z', represented as follows:

A->X
B->Y
C->Z

[0032] Further, the DFM 203 may utilize a query generator to generate a query dynamically based on column mapping metadata from the source columns to target

columns. The DFM 203 may generate the query to be run on the source system 101 with respect to the target columns. As an example, the query generated may be represented as follows:

select CAST(A as int) as X, B as Y, C as Z from T

[0033] Further, the DFM 203 may select an Apache NiFi template for instantiating a pipeline according to transfer use case. As an example, the Apache NiFi template for transferring dataset from 'A', 'B', and 'C' columns of table T from HIVE system to 'X', 'Y', and 'Z' columns of table TA of MySQL system may be represented as HIVE-MYSQL-REPLACE-COUNT. Further, the DFM 203 may select an appropriate connection within the Nifi cluster 207 for the instantiated pipeline. As an example: this may be represented as

Hive – App1-Hive
MySQL – App2-MYSQL

All the aforesaid values may be filled in the Apache NiFi template.

[0034] Further, the DFM 203 may create a flow in the Nifi cluster 207 based on the selected Apache NiFi template. The flow may be scheduled in the Nifi cluster 207 based on a schedule interval defined in the metadata. Upon execution of the NiFi job, it may be verified whether upstream dataset is accessible and is ready for consumption. After availability of upstream dataset, a scheduler may activate a flow activity within the Nifi cluster 207, when the flow activity is scheduled. Further, upon completion of flow activity, a status may be set as complete, which may further be sent to checkpointing 209. Further, the Nifi cluster 207 may push Inline Data quality of the data generated with target format 109 to the Data Quality Microservice 211.

[0035] **Some of the advantages of the present disclosure are listed below.**

[0036] The present disclosure overcomes the need for automatic generation of working NiFi pipeline within a NiFi cluster based on metadata associated with multiple source systems and target systems with different platforms.

[0037] The present disclosure integrates micro services such as checkpointing and data quality services for determining completeness and correctness of data generated.

[0038] The present disclosure automatically captures metadata of a transfer pipeline between disparate systems, and have a logistics engine such as Apache NiFi, which templatize the solution and instantiate the solution for a specific transfer use case.

[0039] The present disclosure does not require manual intervention for generation of transfer pipelines, which significantly reduces time and cost associated with the data transfer process between source systems and target systems.

[0040] **Fig.3** illustrates a block diagram of an exemplary computer system for implementing embodiments consistent with the present disclosure.

[0041] In an embodiment, the computer system 300 may be used to implement the data flow management system. The computer system 300 may include a central processing unit ("CPU" or "processor") 302 for implementing a Data Flow Manager (DFM). The processor 302 may include at least one data processor for capturing metadata of the transfer pipeline between disparate systems (here referred as a source system and a target system), and templatizing automatic conversion of data from a source format to a target format for instantiation of a NiFi pipeline based on captured metadata. The processor 302 may include specialized processing units such as, integrated system (bus) controllers, memory management control units, floating point units, graphics processing units, digital signal processing units, etc.

[0042] The processor 302 may be disposed in communication with one or more input/output (I/O) devices (312 and 313) via I/O interface 301. The I/O interface 301 employ communication protocols/methods such as, without limitation, audio, analog, digital, monoaural, radio corporation of America (RCA) connector, stereo, IEEE-1394 high speed serial bus, serial bus, universal serial bus (USB), infrared, personal system/2 (PS/2) port, bayonet neill-concelman (BNC) connector, coaxial, component, composite, digital visual interface (DVI), high-definition multimedia interface (HDMI), radio frequency (RF) antennas, S-Video, video graphics array (VGA), IEEE 802.11b/g/n/x, Bluetooth, cellular e.g., code-division multiple access (CDMA), high-

speed packet access (HSPA+), global system for mobile communications (GSM), long-term evolution (LTE), worldwide interoperability for microwave access (WiMax), or the like, etc.

[0043] Using the I/O interface 301, the computer system 300 may communicate with one or more I/O devices such as input devices 312 and output devices 313. For example, the input devices 312 may be an antenna, keyboard, mouse, joystick, (infrared) remote control, camera, card reader, fax machine, dongle, biometric reader, microphone, touch screen, touchpad, trackball, stylus, scanner, storage device, transceiver, video device/source, etc. The output devices 313 may be a printer, fax machine, video display (e.g., cathode ray tube (CRT), liquid crystal display (LCD), light-emitting diode (LED), plasma, plasma display panel (PDP), organic light-emitting diode display (OLED) or the like), audio speaker, etc.

[0044] In some embodiments, the processor 302 may be disposed in communication with a communication network 309 via a network interface 303. The network interface 303 may communicate with the communication network 309. The network interface 303 may employ connection protocols including, without limitation, direct connect, ethernet (e.g., twisted pair 10/100/1000 Base T), transmission control protocol/internet protocol (TCP/IP), token ring, IEEE 802.11a/b/g/n/x, etc. The communication network 309 may include, without limitation, a direct interconnection, local area network (LAN), wide area network (WAN), wireless network (e.g., using Wireless Application Protocol), the Internet, etc. Using the network interface 303 and the communication network 309, the computer system 300 may communicate with a database 314, which may be the enrolled templates database 113. The network interface 303 may employ connection protocols include, but not limited to, direct connect, ethernet (e.g., twisted pair 10/100/1000 Base T), transmission control protocol/internet protocol (TCP/IP), token ring, IEEE 802.11a/b/g/n/x, etc.

[0045] The communication network 309 includes, but is not limited to, a direct interconnection, a peer to peer (P2P) network, local area network (LAN), wide area network (WAN), wireless network (e.g., using Wireless Application Protocol), the Internet, Wi-Fi and such. The communication network 309 may either be a dedicated

network or a shared network, which represents an association of the different types of networks that use a variety of protocols, for example, hypertext transfer protocol (HTTP), transmission control protocol/internet protocol (TCP/IP), wireless application protocol (WAP), etc., to communicate with each other. Further, the communication network 309 may include a variety of network devices, including routers, bridges, servers, computing devices, storage devices, etc.

[0046] In some embodiments, the processor 302 may be disposed in communication with a memory 305 (e.g., RAM, ROM, etc. not shown in FIGURE 3) via a storage interface 304. The storage interface 304 may connect to memory 305 including, without limitation, memory drives, removable disc drives, etc., employing connection protocols such as, serial advanced technology attachment (SATA), integrated drive electronics (IDE), IEEE-1394, universal serial bus (USB), fiber channel, small computer systems interface (SCSI), etc. The memory drives may further include a drum, magnetic disc drive, magneto-optical drive, optical drive, redundant array of independent discs (RAID), solid-state memory devices, solid-state drives, etc.

[0047] The memory 305 may store a collection of program or database components, including, without limitation, user interface 306, an operating system 307, etc. In some embodiments, computer system 300 may store user/application data, such as, the data, variables, records, etc., as described in this disclosure. Such databases may be implemented as fault-tolerant, relational, scalable, secure databases such as Oracle or Sybase.

[0048] The operating system 307 may facilitate resource management and operation of the computer system 300. Examples of operating systems include, without limitation, Apple™ Macintosh ™ OS X™, UNIX™, Unix-like system distributions (e.g., Berkeley Software Distribution (BSD), FreeBSD™, Net BSD™, Open BSD™, etc.), Linux distributions (e.g., Red Hat™, Ubuntu™, K-Ubuntu™, etc.), International Business Machines (IBM™) OS/2™, Microsoft Windows™ (XP™, Vista/7/8, etc.), Apple iOS™, Google Android™, Blackberry™ operating system (OS), or the like. In some embodiments, the computer system 300 may implement web browser 308 stored program components. Web browser 308 may be a hypertext viewing application, such

as Microsoft™ Internet Explorer™, Google Chrome™, Mozilla Firefox™, Apple™ Safari™, etc. Secure web browsing may be provided using secure hypertext transport protocol (HTTPS), secure sockets layer (SSL), transport layer security (TLS), etc. Web browsers 308 may utilize facilities such as AJAX, DHTML, Adobe™ Flash, Javascript, Application Programming Interfaces (APIs), etc.

[0049] According to some non-limiting embodiments or aspects, a computer program product including at least one non-transitory computer-readable medium including one or more instructions.

[0050] The illustrated steps are set out to explain the exemplary embodiments shown, and it should be anticipated that ongoing technological development will change the manner in which particular functions are performed. These examples are presented herein for purposes of illustration, and not limitation. Further, the boundaries of the functional building blocks have been arbitrarily defined herein for the convenience of the description. Alternative boundaries can be defined so long as the specified functions and relationships thereof are appropriately performed. Alternatives (including equivalents, extensions, variations, deviations, etc., of those described herein) will be apparent to persons skilled in the relevant art(s) based on the teachings contained herein. Such alternatives fall within the scope and spirit of the disclosed embodiments. Also, the words "comprising," "having," "containing," and "including," and other similar forms are intended to be equivalent in meaning and be open ended in that an item or items following any one of these words is not meant to be an exhaustive listing of such item or items or meant to be limited to only the listed item or items. It must also be noted that as used herein, the singular forms “a,” “an,” and “the” include plural references unless the context clearly dictates otherwise.

[0051] Furthermore, one or more computer-readable storage media may be utilized in implementing embodiments consistent with the present disclosure. A computer readable storage medium refers to any type of physical memory on which information or data readable by a processor may be stored. Thus, a computer readable storage medium may store instructions for execution by one or more processors, including instructions for causing the processor(s) to perform steps or stages consistent with the

embodiments described herein. The term “computer readable medium” should be understood to include tangible items and exclude carrier waves and transient signals, i.e., are non-transitory. Examples include random access memory (RAM), read-only memory (ROM), volatile memory, nonvolatile memory, hard drives, CD ROMs, DVDs, flash drives, disks, and any other known physical storage media.

[0052] Finally, the language used in the specification has been principally selected for readability and instructional purposes, and it may not have been selected to delineate or circumscribe the inventive subject matter. Accordingly, the disclosure of the embodiments of the disclosure is intended to be illustrative, but not limiting, of the scope of the disclosure.

METHOD AND SYSTEM FOR MANAGING DATA FLOW BETWEEN SOURCE SYSTEM AND TARGET SYSTEM

ABSTRACT

The present disclosure relates to a method and a system (103) for managing data flow between a source system (101) and a target system (105). Here, the system comprises a Metadata Definer (MD) 201 for defining metadata of the input dataset associated with the source system (101), output dataset associated with the target system (105), the data transfer pipeline between the input and output datasets and its internal transformation. The system further comprises a Data Flow Manager (DFM) (203) which extracts metadata input from the MD 201, and determines source type, target type, transfer type, data quality strategy, frequency of pipeline, and column mapping from source to target based on the metadata. Further, the DFM 203 generates query to be run on the source system (101) to obtain records (113,117) in target format (109) at target system (105), selects a template for instantiating pipeline. Thus, the present disclosure enables metadata driven data transfer capability between disparate systems which is cost effective and time efficient, by creating an automated pipeline without requiring manual intervention.

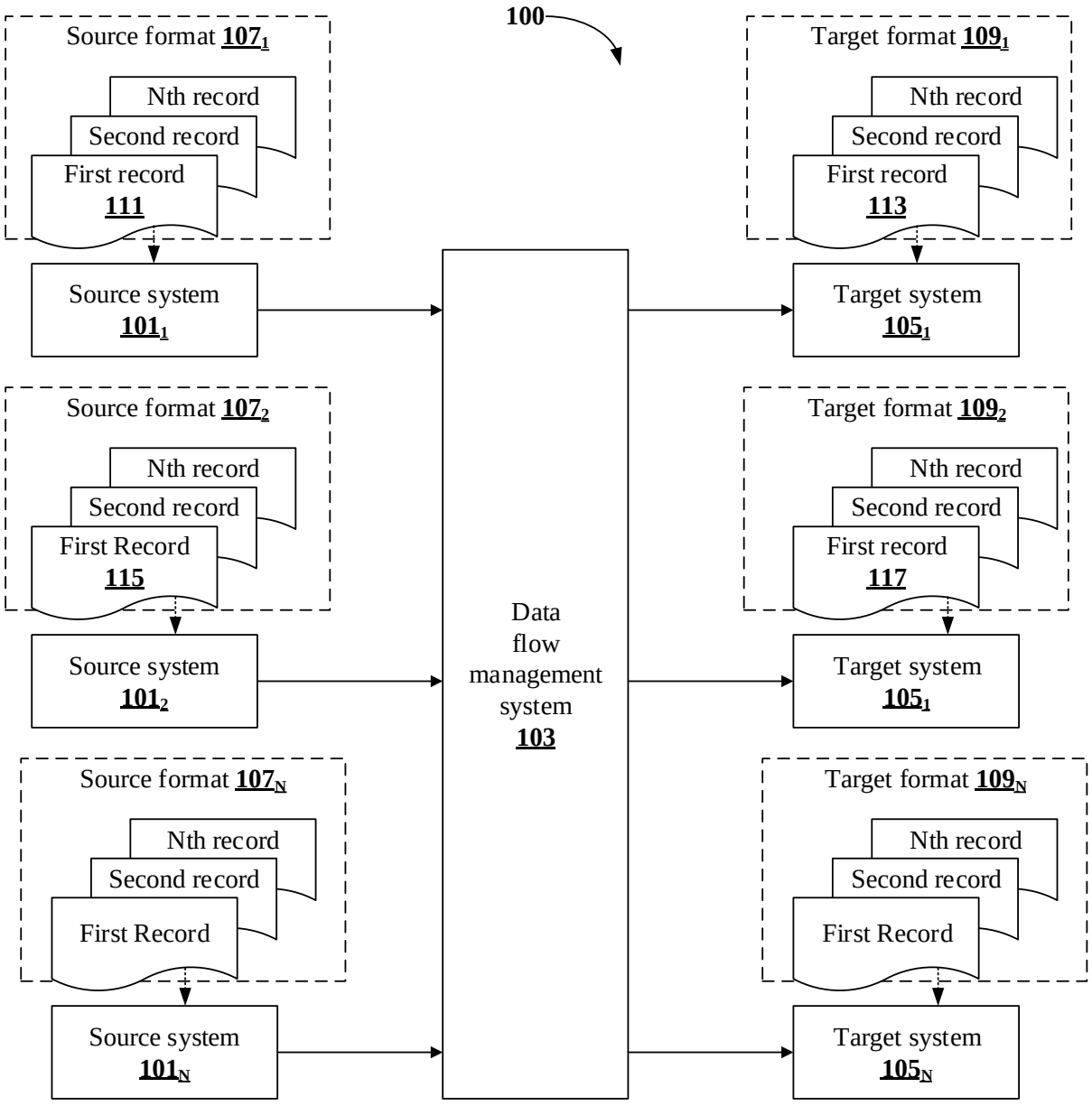
100
Source format 107₁
Nth record
Second record
First record
111
Source system
101₁
Target format 109₁
Nth record
Second record
First record
113
Target system
105₁
Source format 107₂
Nth record
Second record
First Record
115
Source system
101₂
Data
flow
management
system
103
Target format 109₂
Nth record
Second record
First record
117
Target system
105₁
Source format 107N
Nth record
Second record
First Record
Source system
101N
Target format 109N
Nth record
Second record
First Record
Target system
105N

Fig. 1

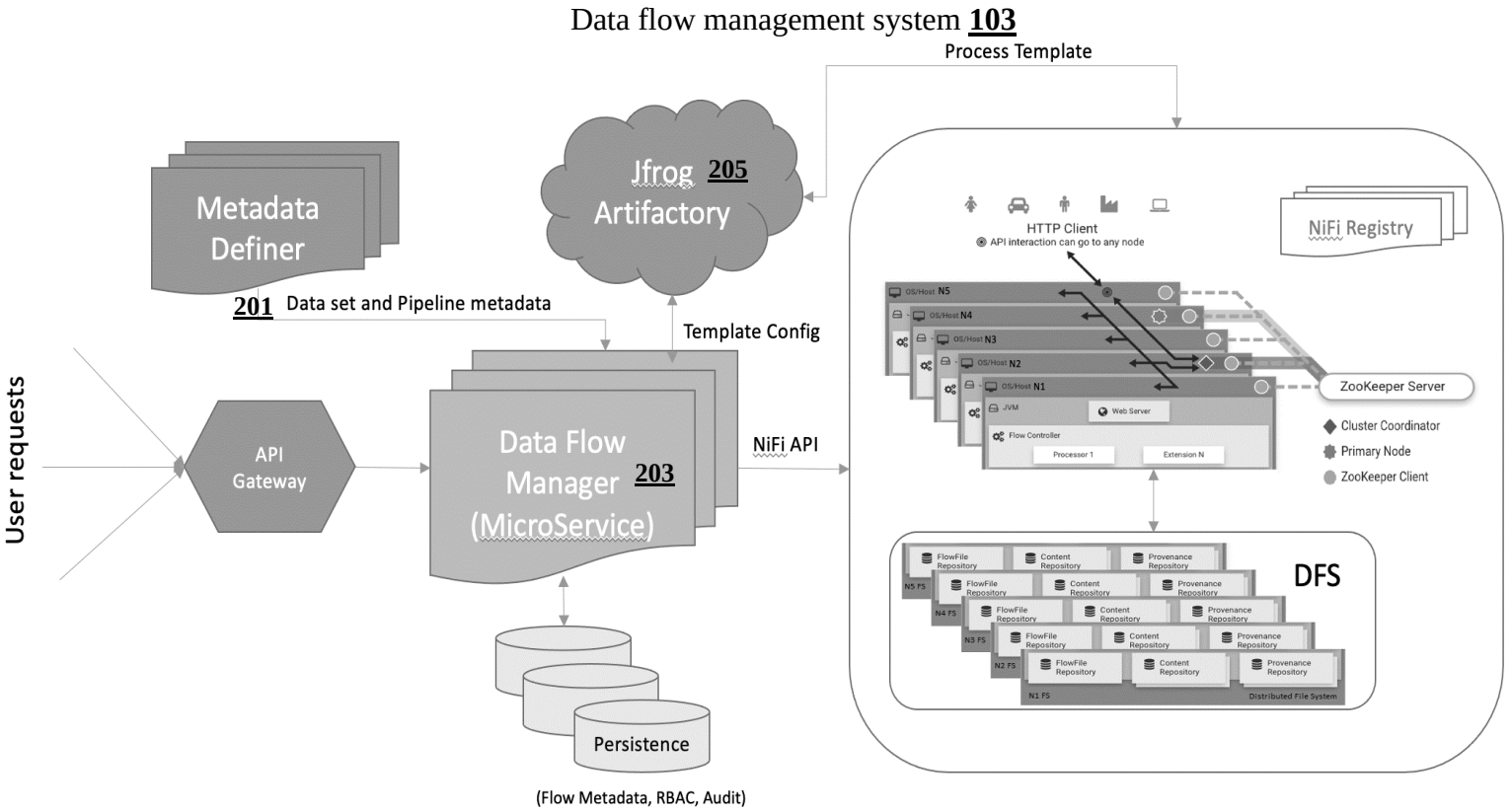
Data flow management system 103
Process Template
Jfrog 205
Artifactory
Metadata
Definer
201 Data set and Pipeline metadata
Template Config
User requests
API
Gateway
Data Flow
Manager 203
(MicroService)
NiFi API
Persistence
(Flow Metadata, RBAC, Audit)
HTTP Client
API interaction can go to any node
NiFi Registry
ZooKeeper Server
Cluster Coordinator
Primary Node
ZooKeeper Client
DFS
Distributed File System

Fig.2a

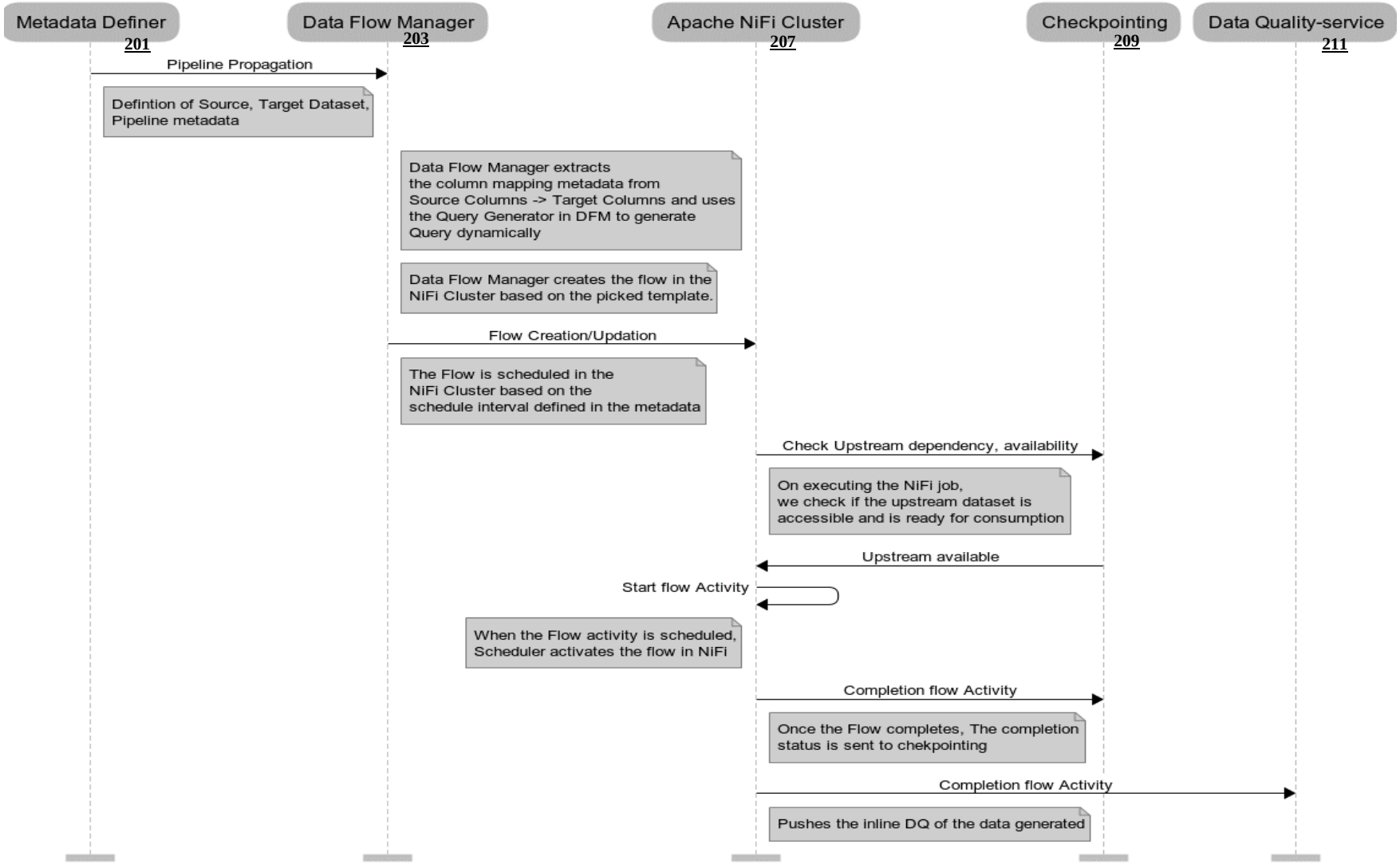
Metadata Definer
201
Data Flow Manager
203
Apache NiFi Cluster
207
Checkpointing
209
Data Quality-service
211
Pipeline Propagation
Defintion of Source, Target Dataset, Pipeline metadata
Data Flow Manager extracts the column mapping metadata from Source Columns -> Target Columns and uses the Query Generator in DFM to generate Query dynamically
Data Flow Manager creates the flow in the NiFi Cluster based on the picked template.
Flow Creation/Updation
The Flow is scheduled in the NiFi Cluster based on the schedule interval defined in the metadata
Check Upstream dependency, availability
On executing the NiFi job, we check if the upstream dataset is accessible and is ready for consumption
Upstream available
Start flow Activity
When the Flow activity is scheduled, Scheduler activates the flow in NiFi
Completion flow Activity
Once the Flow completes, The completion status is sent to chekpointing
Completion flow Activity
Pushes the inline DQ of the data generated

Fig.2b

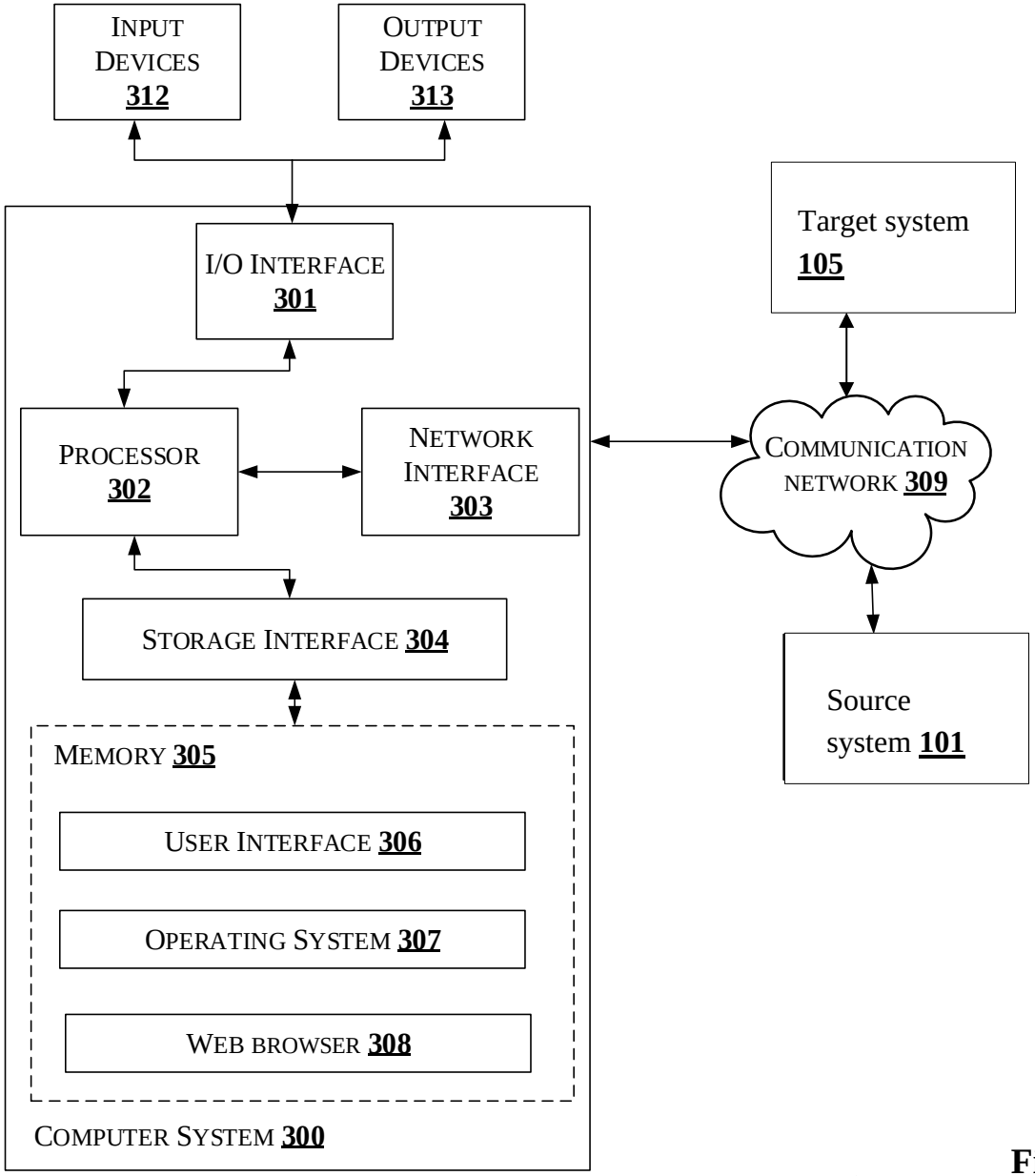
INPUT DEVICES 312
OUTPUT DEVICES 313
I/O INTERFACE 301
PROCESSOR 302
NETWORK INTERFACE 303
STORAGE INTERFACE 304
MEMORY 305
USER INTERFACE 306
OPERATING SYSTEM 307
WEB BROWSER 308
COMPUTER SYSTEM 300
Target system 105
COMMUNICATION NETWORK 309
Source system 101

Fig.3